Foundations for Generalized Planning in Unbounded Stochastic Domains*

Vaishak Belle
KU Leuven
vaishak@cs.kuleuven.be

Hector J. Levesque
University of Toronto
hector@cs.toronto.edu

Abstract

Generalized plans, such as plans with loops, are widely used in AI. Among other things, they are straightforward to execute, they allow action repetition, and they solve multiple problem instances. However, the correctness of such plans is non-trivial to define, making it difficult to provide a clear specification of what we should be looking for. Proposals in the literature, such as strong planning, are universally adopted by the community, but were initially formulated for finite state systems. There is yet to emerge a study on the sensitivity of such correctness notions to the structural assumptions of the underlying plan framework.

In this paper, we are interested in the applicability and correctness of generalized plans in domains that are possibly unbounded, and/or stochastic, and/or continuous. To that end, we introduce a generic controller framework to capture different types of planning domains. Using this framework, we then study a number of termination and goal satisfaction criteria from first principles, relate them to existing proposals, and show plans that meet these criteria in the different types of domains.

1 Introduction

Consider the problem of chopping a tree. If we are told that a *chop* action reduces the thickness of the tree by a unit, and that the agent can sense whether the tree is still standing, the plan structure in Figure 1 is a simple and compact controller that achieves the objective, requiring no internal memory. Intuitively, the controller is reasonable for a tree of any thickness, because (a) the controller will execute as many actions as needed to bring down the tree (*correctness*), and (b) the controller will stop thereafter (*termination*).

Automated planning is a central concern in high-level symbolic AI research, with applications in logistics, robotics and service composition. In the simple case of an agent operating in a known world, the output of a planner is a sequence of actions to be performed. In an incompletely known world, an agent can appeal to its many sensors, and so what is expected is a *conditional plan* that branches on sensing outcomes. The attractiveness, then, of iterative/loopy plan structures like the one in Figure 1 is threefold: (a) their memoryless nature is ideal for systems with limited resources (*e.g.* mobile robots (Matarić 2007)), (b) they allow action repetition, as needed in the presence of nondeterminism, and (c) they behave like conditional plans for a large (possibly infinite) number of problem instances (*e.g.* tree thickness). While early work in this area identified major computational challenges (Manna and Waldinger 1980; Biundo 1994; Stephan and Biundo 1996), significant progress has been made on synthesizing loopy plans in recent years (Cimatti et al. 2003; Levesque 2005; Bonet, Palacios, and Geffner 2009; Srivastava 2010; Hu and De Giacomo 2013).

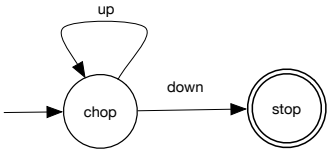

Figure 1: controller for the tree chop problem

Unfortunately, the correctness of such generalized plans is non-trivial to define: that is, what precisely are they generalizing and in which sense are they reasonable for a planning problem? In the absence of a precise specification, it is difficult to identify what we should be looking for.

An early proposal due to Levesque (1996) argued that such a plan should be tested for termination and correctness for all possible initial states of a planning problem. Although formulated in the expressive language of the situation calculus (Reiter 2001), nondeterministic outcomes for actions was not considered. In that vein, Cimatti et al. (2003) later argued that there are conceptual difficulties in understanding the correctness of plans when actions have nondeterministic outcomes. They defined the notions of *weak, strong* and *strong cyclic* solutions formulated in terms of action histories that reach the goal state. An informal probabilistic interpretation for these notions was also suggested: weak plans, for example, reach the goal with a non-zero probability. Nonetheless, although nondeterminism is addressed in their work, Cimatti et al. assume a finite state system.

Despite the almost universal adoption of these notions

*We thank the anonymous reviewers for helpful comments. The first author is supported by the Research Foundation-Flanders (FWO-Vlaanderen).

in the planning community today (Hu and Levesque 2011; Bonet and Geffner 2015; Srivastava et al. 2015; Camacho et al. 2015), there is yet to emerge a study on the sensitivity of such correctness notions to the structural assumptions (*e.g.* determinism, finiteness) of the underlying plan framework. Consider, for example, that nondeterministic outcomes are most prevalent in robotic applications, where the noise of effectors is often characterized by continuous probability distributions (Thrun, Burgard, and Fox 2005), that is, there are uncountably many outcomes. Consider that in many real-world applications, fluents typically take values from infinite or uncountable sets (*e.g.* resources, time). In these scenarios, how are we to define categorical notions of termination and goal satisfaction? Under which conditions (if any) are categorical claims unachievable, warranting a compromise in terms of probabilistic notions?

In this paper, we attempt to answer such questions. More broadly, we are interested in the applicability and correctness of generalized plans in domains that are possibly unbounded, and/or stochastic, and/or continuous. To that end, we develop and motivate a new controller framework that handles probabilistic nondeterminism and unbounded state spaces in a general way. Using this framework, we then study a number of termination and goal satisfaction criteria from first principles, relate them to existing proposals such as (Cimatti et al. 2003), and discuss their properties in domains ranging from deterministic ones to those with continuous distributions. The idea then is that a plan can be argued to be correct in terms of some of these criteria. We will discuss sample plans that meet such specifications in a manner that is independent of plan generation.

At the outset, what we will not be doing in this paper is proposing a new planning procedure. We believe that existing procedures like the ones presented in (Levesque 2005; Srivastava 2010; Bonet, Palacios, and Geffner 2009; Hu and De Giacomo 2013) will be enough. What is additionally needed, however, are ways to automatically prove that they are correct for a new set of specifications. We leave this for the future. Our framework, at least, will make clear in which sense plans like the one in Figure 1 are correct in unbounded stochastic domains.

2 A Controller Framework

We are interested in the adequacy of program-like plans that work in multiple, possibly infinitely many, domain instances. We believe these notions of adequacy do not depend on the details of how the planning problem is formalized. Building on the work of (Lin and Levesque 1998), we introduce an abstract framework for our purposes.

The idea is to develop a *specification* of a (robot) controller operating in a dynamic environment, which completely abstracts from the syntactic or structural characterization of a plan representation. The controller encapsulates the actions that some putative agent performs, and the environment is the world in which these actions occur. We will not want to assume, however, that the controller has access to one form of information or another. So from the controller's perspective, the only feedback from the environment are observations received after every action.

Formally, imagine a possibly infinite set of actions $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots\}$ that are parameterless, and a special symbol *stop* not in $\mathcal{A}$ to denote termination. We also imagine a possibly infinite set of observations $\mathcal{O} = \{\beta_1, \beta_2, \ldots\}$ that are parameterless. The controller, then, simply responds to accumulated sensor readings by advising an action:

Definition 1: A controller $\mathcal{C}$ is any function from $\mathcal{O}^*$ to $\mathcal{A} \cup \{stop\}$.

After an action, the environment changes one state to another, and responds with an observation:

Definition 2: An *environment* $\mathcal{E}$ is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega \rangle$:

- $\mathcal{S}$ is a possibly infinite set of states;
- $\mathcal{A}$ is a possibly infinite set of actions (as above);
- $\mathcal{O}$ is a possibly infinite set of observations (as above);
- $\Delta \in [\mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^{\geq 0}]$ is a probabilistic transition function: that is, on doing α at s, $\mathcal{E}$ changes to s' with a likelihood of $\Delta(s, \alpha, s')$;
- $\Omega \in [\mathcal{S} \to \mathcal{O}]$ is a sensor model.

Put together, a *system* encapsulates a controller operating in an environment:

Definition 3: A *system* is a pair $(\mathcal{C}, \mathcal{E})$, where $\mathcal{C}$ is a controller, and $\mathcal{E}$ is an environment.

The dynamics of a system is characterized by its *runs*:

Definition 4: A *history* σ of the system $(\mathcal{C}, \mathcal{E})$ is an element of the set $\mathcal{S}^*$. The *final state* of $\sigma = s_0 \cdots s_n$, written $end(\sigma)$, is s_n, and the *sensing outcomes* for σ, written $sensed(\sigma)$, is $\Omega(s_0) \cdots \Omega(s_n)$. For the empty history ϵ, $end(\epsilon) = sensed(\epsilon) = \epsilon$.

A non-empty history σ is a *run* at a state s of a system $(\mathcal{C}, \mathcal{E})$ if, inductively, either σ is s, or $\sigma = \sigma' \cdot s'$ such that σ' is a run of the system at s, $\mathcal{C}(sensed(\sigma')) = \alpha$ and $\Delta(end(\sigma'), \alpha, s') > 0$. The run σ is said to be *terminating* if $\mathcal{C}(sensed(\sigma)) = stop$. The run σ is said to be *extendable* if there is a non-empty history σ' such that $\sigma \cdot \sigma'$ is also a run at s of $(\mathcal{C}, \mathcal{E})$. A run σ is said to be *undefined* if $\mathcal{C}(sensed(\sigma)) \neq stop$ and σ is not extendable. A *proper* system is one without undefined runs.

The picture is this: starting with state s, on reading $\beta_1 = \Omega(s)$, the robot controller chooses to perform $\alpha_1 = \mathcal{C}(\beta_1)$. The environment probabilistically changes to some s' provided $\Delta(s, \alpha_1, s') > 0$ and returns $\beta_2 = \Omega(s')$ to the controller. The controller now advises $\alpha_2 = \mathcal{C}(\beta_1 \cdot \beta_2)$, and so on, until the controller says *stop*. In this work, we implicitly assume that systems are proper: that is, given a system $(\mathcal{C}, \mathcal{E})$ and any state s of $\mathcal{E}$, there are no undefined runs at s.[1]

Finally, we turn to the notion of a basic and generalized planning problem:

Definition 5: For any $\mathcal{E} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega \rangle$, a *basic planning problem* $\mathcal{P}$ is a tuple $\langle s_0, \mathcal{E}, \mathcal{G} \rangle$ where $s_0 \in \mathcal{S}$ is the initial state and $\mathcal{G} \subseteq \mathcal{S}$ are goal states.

[1] Systems with undefined runs will need to abort before the end of computation, a complication we ignore.

Definition 6: A *generalized planning problem* $\overline{\mathcal{P}}$ is a (possibly infinite) set of basic problems $\{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$, where $\mathcal{P}_i$ agree on everything except the initial state.

Types of Environments

It will be useful to distinguish between types of environments that vary in the nature of the state space or the transition function, for example. We follow this terminology:

Definition 7: Suppose $\mathcal{E} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega \rangle$. Then:

- $\mathcal{E}$ is *fully observable* when $\mathcal{O} = \mathcal{S}$ and Ω is such that for every $s \in \mathcal{S}$, $\Omega(s) = s$.
- $\mathcal{E}$ is *finite* if $\mathcal{S}$ is finite; otherwise $\mathcal{E}$ is *infinite.*
- $\mathcal{E}$ is *noise-free* (or *deterministic*) if for every s and α, there is a unique s' such that $\Delta(s, \alpha, s') = 1$ and for $s'' \neq s'$, $\Delta(s, \alpha, s'') = 0$; otherwise $\mathcal{E}$ is *noisy.*
- $\mathcal{E}$ is *discrete* if for every $(s, \alpha) \in \mathcal{S} \times \mathcal{A}$, $\Delta(s, \alpha, \cdot)$ is a probability mass function. If $\Delta(s, \alpha, \cdot)$ is a probability density function, then $\mathcal{E}$ is *continuous*.[2]

The terminology is extended for a basic planning problem $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ in terms of its environment $\mathcal{E}$. For example, if $\mathcal{E}$ is noise-free, then we say that $\mathcal{P}$ is a noise-free basic planning problem, or simply noise-free problem for short.

Effective Controllers

While our controller formulation was general, we are naturally interested in algorithms of one kind or another that generate plans (that is, controllers that are computable), for which we define:

Definition 8: A controller $\mathcal{C}$ is *effective* if the function $\mathcal{C}$ is recursive.

One simple candidate for an effective controller requiring no internal memory, seen in Figure 1, is the following:

Definition 9: Suppose $acts \subseteq \mathcal{A}$ is a finite set of actions, and $obs \subseteq \mathcal{O}$ a finite set of observations. A *finite memoryless plan* $\mathcal{X}$ is a tuple $\langle \mathcal{Q}, q_0, \gamma, \delta \rangle$:

- $\mathcal{Q}$ is a finite set of control states;
- $q_0 \in \mathcal{Q}$ is the initial control state;
- $\gamma \in [\mathcal{Q} \to (acts \cup \{stop\})]$ is a labeling function;
- $\delta \in [\mathcal{Q}' \times obs \to \mathcal{Q}]$ is a transition function, where $\mathcal{Q}' = \{q \in \mathcal{Q} \mid \gamma(q) \neq stop\}$.

For any environment $\mathcal{E}$, a finite memoryless plan $\mathcal{X}$ can be viewed as a controller $\mathcal{C}$ that behaves as follows for a run $\sigma = s_0 \cdots s_n$ of the system $(\mathcal{C}, \mathcal{E})$:

$$\mathcal{C}(\Omega(s_0) \cdots \Omega(s_n)) = \gamma(q(\Omega(s_0) \cdots \Omega(s_n))))$$

where, $q(\Omega(s_0) \cdots \Omega(s_n)))$ is defined inductively:

$$= \begin{cases} q_0 & \text{if } n = 0 \\ \delta(q(\Omega(s_0) \cdots \Omega(s_{n-1})), \Omega(s_n)) & \text{otherwise} \end{cases}$$

Such controllers are prominent in the literature on generalized planning (Bonet, Palacios, and Geffner 2009; Hu and Levesque 2011; Hu and De Giacomo 2013), and are essentially Moore machines (Hopcroft and Ullman 1979).

Results investigated in the sequel are general, but our discussions will make use of such memoryless plans owing to their simplicity.

3 Adequacy

To formally define the reasonableness of controllers for a problem specification, we will identify various notions of *adequacy*, all of which are characterized in terms of runs. Roughly, if the runs of a system $(\mathcal{C}, \mathcal{E})$ satisfy a property θ (*e.g.* there is a terminating run), we say that $\mathcal{C}$ is a θ-adequate controller. In this section, the notions of adequacy are *categorical* concepts that do not mention probabilities. (Probabilistic notions are discussed in subsequent sections.)

Henceforth, for any $\mathcal{C}$, in the context of a basic problem $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$, we will often simply speak of *runs* of the system $(\mathcal{C}, \mathcal{E})$ with the understanding that these runs are at s_0. Also, the length of a history σ is defined inductively: the length of ϵ is 0, and the length of $\sigma' \cdot s$ is 1 plus the length of σ'. With that, let us begin by stating how generalized planning problems are solved for some notion of adequacy:

Definition 10: Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is a basic planning problem, and $\mathcal{C}$ is any controller. We say $\mathcal{C}$ is θ-adequate for $\mathcal{P}$ if the runs (at s_0) of $(\mathcal{C}, \mathcal{E})$ satisfies θ.

Definition 11: We say a controller $\mathcal{C}$ is θ-adequate for a generalized planning problem $\overline{\mathcal{P}}$ iff $\mathcal{C}$ is θ-adequate for every basic problem in $\overline{\mathcal{P}}$.

Thus, controllers for generalized problems are adequate (in whichever sense) for all of the contained basic problems.[3]

We are now prepared to define our notions of adequacy:

ONE There is a terminating run σ such that $end(\sigma) \in \mathcal{G}$.

PC For every terminating run σ, we have $end(\sigma) \in \mathcal{G}$.

TER For every run σ, there is a history σ' such that $\sigma \cdot \sigma'$ is a terminating run.

BND Every run is *bounded*, that is, there is a number $n \in \mathbb{N}$ such that there is no run of length $> n$.

ACYC If $s \cdots s'$ is a run then $s \neq s'$.

Intuitively, **ONE** says that there is a run to the goal; **PC** denotes *partial correctness,* that is, every terminating run stops at a goal state; **TER** denotes *termination*, that is, it ensures that every run can be extended to a terminating one; **BND** denotes *boundedness*, that is, it disallows runs of arbitrary length; finally, **ACYC** denotes *acyclicity*, that is, it disallows loops.

For the main results of this section, we study some key properties of these notions for the various types of problems:

[2] For ease of presentation, we will not treat environments with both discrete and continuous transition models.

[3] In other words, our results on adequacy apply to basic planning problems even outside the context of generalized planning.

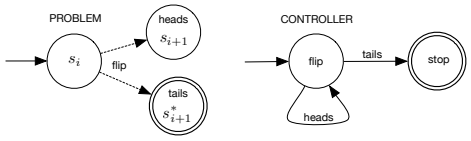

Figure 2: coin flip problem (double stroked circles indicate goal states) and its controller

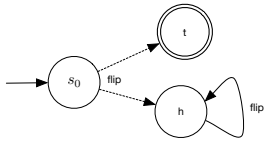

Figure 3: coin flip problem with bad flips

Theorem 12: *In any (basic planning) problem, if* **BND** *holds then* **TER** *holds.*

Proof: By assumption, all runs have length $\leq n$, for some $n \in \mathbb{N}$. In other words, there cannot be a run that is extendable to one of arbitrary length. Let Γ be the set of all runs that are not extendable. (For all runs $\sigma \notin \Gamma$, there must be a history σ' such that $\sigma \cdot \sigma' \in \Gamma$ because σ cannot be extended arbitrarily by assumption.) Since none of the runs $\sigma \in \Gamma$ can be undefined (recall that systems are implicitly assumed to be proper), they must be terminating. ∎

Proposition 13: *In noise-free problems, if there is a run of length k then it is unique.*

Proof: By induction on k. Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is a noise-free basic planning problem, and $\mathcal{C}$ any controller. Clearly there is only one run of length 1, namely s_0. Suppose σ is a run of length k; by hypothesis it is unique. Suppose $\mathcal{C}(sensed(\sigma)) = stop$, then we are done. Suppose $\mathcal{C}(sensed(\sigma)) = \alpha \in \mathcal{A}$. In noise-free environments, there is a unique s' such that $\Delta(end(\sigma), \alpha, s') \neq 0$ and so, $\sigma \cdot s'$ is a run of length $k+1$ and it is unique. ∎

Theorem 14: *In noise-free problems, if* **TER** *holds then* **BND** *holds. This is not the case in noisy ones.*

Proof: Since **TER** holds, for any run σ, by Proposition 13, there is a unique (possibly empty) history σ' such that $\sigma \cdot \sigma'$ is terminating. Clearly, **BND** holds, as there cannot be a run of length greater than that of $\sigma \cdot \sigma'$.

For noisy domains, we provide a counterexample in Figure 2. Imagine an environment with states $s_0, s_1, s^*_1, \ldots,$ such that from state s_i, a coin flip action nondeterministically changes the environment to either s_{i+1} where heads is observed, or to s^*_{i+1} where tails is observed. Suppose a basic planning problem is specified by letting s_0 be the initial state and s^*_i be the goal states, that is, the goal is to obtain a tails. For this problem, given any run $\sigma = s_0 \cdots s_n$ that is not terminating, clearly there is a history, namely s^*_{n+1}, such that $\sigma \cdot s^*_{n+1}$ is a terminating run. So **TER** indeed holds, but **BND** does not since the length of σ can be arbitrary. ∎

Theorem 15: *In noise-free problems,* **ONE** *is equivalent to* $\{$**PC**, **TER**$\}$. *This is not the case in noisy problems.*

Proof: For noise-free problems, suppose **ONE** holds: there is a terminating run σ, say of length k, and $end(\sigma) \in \mathcal{G}$. By Proposition 13, σ is the only run of length k, *i.e.* every other run $\sigma' \neq \sigma$ can be extended to σ, and by assumption, it is terminating and so **TER** holds. By that account, σ is also the only terminating run and so **PC** holds. Conversely, **TER** ensures that there is a terminating run and by Proposition 13, **PC** ensures that this reaches the goal, and so **ONE** is true.

For noisy problems, we provide a counterexample in Figure 3. Consider an environment $\mathcal{E}$ with states $\{s_0, h, t\}$ and a basic problem $\langle s_0, \mathcal{E}, \{t\} \rangle$. A coin flip nondeterministically changes the environment from s_0 to either h where heads is observed, or to t where tails is observed. No actions are possible at t. Coin flips are possible at h, but the environment changes to h itself. In other words, if tails is not observed for the first coin flip, then it will not be observed for all coin flips. Using the controller from Figure 2, we see that **ONE** holds by getting a tails on the first flip. But **TER** does not hold because there is no history that makes the run $s_0 \cdot h$ a terminating one. ∎

Theorem 16: *In finite problems, if* **ACYC** *holds then* **BND** *holds. This is not the case in infinite ones.*

Proof: In finite problems, $\mathcal{S}$ is finite. Suppose **ACYC** holds but **BND** does not. Then there is a run of length $> |\mathcal{S}|$, which means that by the pigeon hole principle, the run must mention some state twice. Contradiction.

If $\mathcal{S}$ is infinite, a system can be easily constructed for which a run visits every state exactly once, and so **ACYC** would hold but **BND** would not. The coin flip problem in Figure 2 is one such example: runs of the form $s_0 \cdot s_1 \cdots s_n$ never visit the same state twice but are unbounded. ∎

In Section 7, we revisit these results and relate them to existing proposals on plan correctness.

4 Discrete Probabilistic Adequacy

The categorical notions of termination and correctness discussed thus far disregarded the actual probabilities of the transitions in the environment. It is then natural to investigate whether a characterization of adequacy can be developed that accounts for these probabilities. More importantly, does categorical adequacy imply probabilistic adequacy? This section studies such a characterization for discrete problems, and answers in the affirmative. (Continuous problems are deferred to the subsequent section.)

We begin by applying transition probabilities to runs of arbitrary length. We define the *likelihood* of a run σ at s in the system $(\mathcal{C}, \mathcal{E})$, denoted $l(\sigma)$, inductively:

- if $\sigma = s$, then $l(s) = 1$;
- if $\sigma = \sigma' \cdot s'$ and $\mathcal{C}(sensed(\sigma')) = \alpha$, then $l(\sigma) = l(\sigma') \times \Delta(end(\sigma'), \alpha, s')$.

The probabilistic analogue to **TER** for a basic planning problem $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is then defined as:

$$\sum_{\{\sigma | \sigma \text{ is a terminating run}\}} l(\sigma) \qquad \textbf{(LTER)}$$

which accounts for the likelihoods of terminating runs. Likewise, the counterpart to **PC** is defined as:

$$\frac{1}{\eta} \sum_{\{\sigma | \sigma \text{ is a terminating run and } end(\sigma) \in \mathcal{G}\}} l(\sigma) \qquad \textbf{(LPC)}$$

which accounts for the likelihoods of terminating runs that also reach goal states. Here, $\eta \neq 0$ is the normalization factor, and it is naturally defined in terms of terminating runs regardless of whether they reach the goal states. Therefore:

$$\eta = \textbf{LTER}.$$

For the main results of this section, we relate the probabilistic and the categorical accounts below.

Lemma 17: *Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is any discrete problem, and $\mathcal{C}$ any controller. For any $k \in \mathbb{N}$, let*

$$\Gamma_k = \{\sigma \mid \sigma \text{ is a run of length } k\} \cup \{\sigma \mid \sigma \text{ is a terminating run of length } \leq k\}.$$

(As usual, runs at s_0 of $(\mathcal{C}, \mathcal{E})$.) Then $\sum_{\sigma \in \Gamma_k} l(\sigma) = 1$.

Proof: By induction on k. For $k = 1$, the proof is trivial since $l(s_0) = 1$ by definition. Assume claim is true for Γ_k. Clearly $\Gamma_k \cap \Gamma_{k+1}$ is the set of all terminating runs of length $\leq k$. Consider $\Gamma^* = \Gamma_{k+1} - \Gamma_k$. Observe that $\sigma \in \Gamma^*$ only if there is a $\sigma' \in \Gamma_k$ and a history of length 1, say s, such that $\sigma = \sigma' \cdot s$, that is, $\mathcal{C}(sensed(\sigma')) = \alpha$ and $\Delta(end(\sigma'), \alpha, s) \neq 0$. Let $exts(\sigma') = \{\sigma \in \Gamma^* \mid \sigma \text{ is an extension of } \sigma'\}$. Since $\Delta(end(\sigma'), \alpha, \cdot)$ is a probability mass function, $l(\sigma') = \sum_{\sigma \in exts(\sigma')} l(\sigma)$. It then follows that:

$$\sum_{\sigma \in \Gamma^*} l(\sigma) = \sum_{\sigma \in (\Gamma_k - \Gamma_{k+1})} l(\sigma).$$

Using this equivalence in the below equality:

$$\sum_{\sigma \in \Gamma_{k+1}} l(\sigma) = \sum_{\sigma \in (\Gamma_k \cap \Gamma_{k+1})} l(\sigma) + \sum_{\sigma \in \Gamma^*} l(\sigma)$$

we obtain that $\sum_{\sigma \in \Gamma_{k+1}} l(\sigma) = \sum_{\sigma \in \Gamma_k} l(\sigma)$ which is 1 by the hypothesis. ∎

Theorem 18: *In discrete problems, if* **LTER**=1*, then* **TER** *holds.*

Proof: Suppose **LTER** is 1 but **TER** does not hold: there is a run σ^* such that for every σ' we have that $\sigma^* \cdot \sigma'$ is not a terminating run. By definition, $l(\sigma^*) > 0$. (Recall that runs are admitted only by transitions with non-zero probability.) Suppose the length of σ^* is k and consider the set Γ_k as in Lemma 17. By Lemma 17, $\sum_{\sigma \in \Gamma_k} l(\sigma) = 1$ and so $\sum_{\sigma \in \Gamma^*} l(\sigma) < 1$, where $\Gamma^* = \Gamma_k - \{\sigma^*\}$. Thus, for every $n > k$, by assumption, every extension of σ^* of length n that is a run is not terminating, and so $\sum_{\sigma \in \Gamma^{**}} l(\sigma) < 1$, where $\Gamma^{**} = \Gamma_n - exts(\sigma^*)$, Γ_n is as in Lemma 17, and $exts(\sigma^*)$ is the set of all extensions of σ^* that are runs of length n. Contradiction. ∎

Theorem 19: *In discrete problems, if* **LPC**=1*, then* **PC** *holds.*

Proof: Suppose **LPC**=1, that is, **LPC**=**LTER**, but **PC** does not hold: there is a terminating run σ^* that does not reach a goal state. Suppose Γ is the set of terminating runs. By definition, $l(\sigma^*) > 0$, and so clearly:

$$\sum_{\sigma \in \Gamma} l(\sigma) > \sum_{\sigma \in (\Gamma - \{\sigma^*\})} l(\sigma).$$

So **LPC**, which is defined over the runs $\subseteq (\Gamma - \{\sigma^*\})$ reaching a goal state, cannot be equal to **LTER**. Contradiction. ∎

It is easy to show that categorical adequacy implies probabilistic adequacy:

Proposition 20 : *In discrete problems,* **TER** *implies* **LTER**=1*, and* **PC** *implies* **LPC**=1.

Putting this together, we get the following equivalence:

Corollary 21: *In discrete problems,* **LTER**=1 *iff* **TER***, and* **LPC**=1 *iff* **PC**.

5 Continuous Probabilistic Adequacy

In this section, we want to study the consequences of a continuous transition function on probabilistic adequacy. Of course, in continuous problems, the uncountable nature of the state space precludes sums. But readers may observe that **LTER** and **LPC** are not directly amenable to a continuous version, since it is unclear how to integrate over the set of terminating runs that vary arbitrarily in length. So what we attempt first is a reformulation of **LTER** and **LPC** for discrete problems where the sums are defined over the set of states rather than runs. This scheme is then generalized.

Let us define $L(\sigma, n)$ as the sum of the likelihoods of all histories σ' of maximal length n such that $\sigma \cdot \sigma'$ is a terminating run. More precisely:

Definition 22: Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is any discrete problem, $\mathcal{C}$ any controller, and σ any history of $(\mathcal{C}, \mathcal{E})$. Then, define $L(\sigma, n)$ inductively:

- $L(\sigma, 1) = \begin{cases} 1 & \text{if } \sigma \text{ is terminating} \\ 0 & \text{otherwise} \end{cases}$
- $L(\sigma, n+1) = \begin{cases} L(\sigma, n) & \text{if } \sigma \text{ is terminating} \\ succ & \text{otherwise} \end{cases}$

  where,

$$succ \doteq \sum_s L(\sigma \cdot s, n) \times \Delta(end(\sigma), \mathcal{C}(sensed(\sigma)), s)$$

Let us define $L^\bullet(\sigma, n)$ inductively, which is exactly like $L(\sigma, n)$ except for the base case:

$$L^\bullet(\sigma, 1) = \begin{cases} 1 & \text{if } \sigma \text{ is terminating and } end(\sigma) \in \mathcal{G} \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 23: *In discrete problems,* **LTER** *computes the same number as:*

$$\lim_{n \to \infty} L(s_0, n) \qquad (\dagger)$$

Proof: Let $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ be any discrete problem and $\mathcal{C}$ any controller. Let Γ be the set of terminating runs at s_0 of $(\mathcal{C}, \mathcal{E})$. Let $N = \max\{\text{length of } \sigma \mid \sigma \in \Gamma\}$.

For $n \geq 1$, let

$$\mathbf{LTER}_n = \sum_{\{\sigma | \sigma \text{ is a terminating run of length } \leq n\}} l(\sigma)$$

An induction argument will show that $\mathbf{LTER}_n = L(s_0, n)$. So clearly $\mathbf{LTER} = \mathbf{LTER}_N = L(s_0, N)$. By definition, we have $L(s_0, N) = L(s_0, N + k)$ for $k > 0$. Thus the sequence $L(s_0, 1), \ldots, L(s_0, N), \ldots$ converges and is $L(s_0, N)$. Therefore, **LTER**=(†). ∎

Theorem 24: *In discrete problems,* **LPC** *computes the same number as:*

$$\frac{1}{\eta} \lim_{n \to \infty} L^{\bullet}(s_0, n) \qquad (\ddagger)$$

Proof: We need a simple variation of the argument for Theorem 23. We begin by defining Γ to be the set of all terminating runs that also reach a goal state. The rest proceeds analogously. ∎

These are now amenable to immediate generalizations:

Definition 25: In continuous problems, **LTER** is given by (†) and **LPC** by (‡), where $L(\sigma, n)$ and $L^{\bullet}(\sigma, n)$ are as in Definition 22 except that

$$succ \doteq \int_s L(\sigma \cdot s, n) \times \Delta(end(\sigma), \mathcal{C}(sensed(\sigma)), s).$$

In a continuous setting, while a density is accorded to all points of the state space, the probability of any one point (or a finite set of points) is zero. This leads to surprising deviations from Corollary 21.

Theorem 26: *In continuous problems,* **LTER**=1 *does not imply* **TER**.

Proof: Proof by counterexample. The idea is to allow one run to never terminate but which nonetheless does not affect the outcome of integration. Imagine a pick action that nondeterministically changes an environment from s_0 to a state in $\mathcal{S}' = \{s'_x \mid x \in [0, 1]\}$, corresponding to a sample drawn from the interval $[0, 1]$. All states contain a tree and respond with an observation that the tree is up. States in $\mathcal{S}' - \{s'_0\}$ contain a wooden tree that can be brought down by doing some number of chop actions. However, the state s'_0 has a steel tree that can never be brought down; see Figure 4. Here, integrating over $\mathcal{S}' - \{s'_0\}$ yields **LTER**=1, but **TER** does not hold because no extension of $s_0 \cdot s'_0$ will terminate. ∎

Theorem 27: *In continuous problems,* **LPC**=1 *does not imply* **PC**.

Proof: Proof by counterexample. The idea is to allow one run to not reach the goal but which will not affect the outcome of integration. Imagine a robot operating in a 1-dimensional world, at location 0. Its goal is to be any location but at 0 using a single move action. Assuming this action is characterized by a continuous noise model, such as a Gaussian, all outcomes, including the one where the robot moves by 0 units, are given a non-zero density. Assume all states vacuously respond with the observation *true*. A very simple controller can be specified for this problem; see Figure 5. (Note that the argument to the move action will not affect the nature of this example.) So, all runs terminate, but the one where $loc = 0$ is not a goal state. Integrating over the rest still leads to **LPC**=1, but **PC** does not hold. ∎

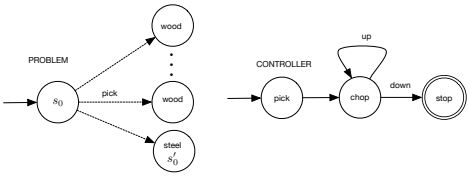

Figure 4: wood and steel trees

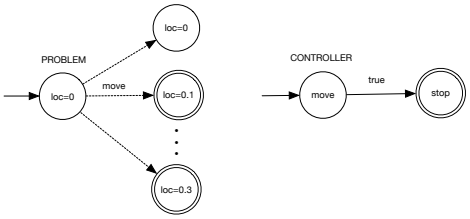

Figure 5: continuous problem with the goal $loc \neq 0$

Proposition 20 is easily shown to hold also in the continuous case:

Proposition 28 : *In continuous problems,* **TER** *implies* **LTER**=1*, and* **PC** *implies* **LPC**=1.

Naturally, this means that equivalences from Corollary 21 do not hold for continuous problems:

Corollary 29: *In continuous problems,* **TER** *and* **LTER**=1 *are not equivalent, and* **PC** *and* **LPC**=1 *are not equivalent.*

6 Barriers to Adequacy

Having investigated relations between termination and correctness criteria across environments, we are now in a position to ask: are there planning problems for which only a particular sort of adequacy can be provided? In this section, we answer positively to that question. We consider a set of planning problems using the inventory of examples dealt with so far, and discuss sample plans. Like with troublesome domains such as Figure 4, it will be easy to argue that any variation of these sample plans can, for example, only provide some flavour of probabilistic adequacy, but never categorical ones.

Consider the following three criteria for termination: **BND**, **LTER**=1 and **LTER** < 1. (We favor **BND**-adequacy over **TER**-adequacy; as seen with Theorem 14, **BND** ensures bounded runs also in infinite problems.) Consider the following three criteria for correctness: **PC**, **LPC**=1 and **LPC** < 1. Below, we enumerate problems that are distinctive for every pair of termination and correctness criteria:

	PC	**LPC**=1	**LPC** < 1
BND	(1)	(2)	(3)
LTER=1	(4)	(5)	(6)
LTER < 1	(7)	(8)	(9)

In a nutshell, columns will share the goal specification, and rows will share a randomness component.

1. *Standard tree chop (Figure 1)*: the agent chops down a tree of thickness $n > 0$ by executing n chop actions. So there is a unique terminating run of length $n + 1$ reaching the goal. Therefore, both **BND** and **PC**.

2. *Point goal (Figure 5)*: all runs $s_0 \cdot s'_x$ for $x \in [0, 1]$ are terminating, and so **BND**. As the goal is $loc \neq 0$, every run other than $s_0 \cdot s'_0$ reach goal states and so **PC** does not hold. Owing to the continuity, **LPC**=1.

3. *Interval goal*: consider the controller and environment from item 2 but assume that the planning problem has as its goal $loc > .5$. As before, all runs $s_0 \cdot s'_x$ for $x \in [0, 1]$ are terminating, and so **BND**. However, every run other than $s_0 \cdot s'_y$ for $y \in [0, .5]$ are goal states; so **PC** does not hold. Moreover, although the probability of any single s'_y is 0 (as in item 2), the probability of this set of points, which comprise an interval, is non-zero. So **LPC** < 1. (As with item 2, the argument to the move action will not affect the nature of this example.)

4. *Flip and chop (Figure 6)*: a tails on a coin flip permits the agent to chop down a tree. All terminating runs reach the goal, and so **PC**, but there is no bound because of the coin flip. So **BND** does not hold, but **TER** and **LTER**=1 hold.

5. *Flip and point goal (Figure 7)*: a tails on a coin flip permits the agent to attempt $loc \neq 0$ using a single move action. As in item 4, owing to the coin flip, there is no bound on the runs, but **TER** and **LTER**=1 hold. Moreover, as established for item 2, **PC** does not hold but **LPC**=1.

6. *Flip and interval goal (Figure 7)*: a tails on a coin flip permits the agent to attempt $loc > .5$ using a single move action. As seen in item 5, **LTER**=1, but **BND** does not hold. Moreover, as established for item 3, **LPC** < 1.

7. *Bad flip and chop*: consider the environment in Figure 8 where a coin flip at s_0 nondeterministically changes to either h, where heads is observed, or to t where tails is observed. The state t also has a tree of unit thickness. The goal is to bring down this tree. Only coin flips are possible at h, but the environment changes to h itself. Only chop actions are possible at t. Using the controller from item 4, we observe that no extension of the run $s_0 \cdot h$ will terminate, and so **TER** does not hold. (Moreover, coin flips means **BND** does not hold.) In other words, **LTER** is determined from the transition probability for t and so is < 1. Nonetheless, all extensions of $s_0 \cdot t$ that are runs are both terminating and reach goal states, and so **PC** holds.

8. *Bad flip and point goal*: consider an environment like the one in Figure 8, except that at t, we imagine a robot at location 0, that is, $loc = 0$. A move action nondeterministically changes the environment to $\{t'_x \mid x \in [0, 1]\}$ in that t'_x is a state where $loc = x$. The goal is to be at a state where $loc \neq 0$. Using the controller from item 5, we observe that, as in item 5, **PC** does not hold but **LPC**=1. Owing to the bad coin flip, from item 7, we obtain **LTER** < 1.

9. *Bad flip and interval goal*: consider the environment from item 8, except that the goal is $loc > .5$. As in item 6, owing to the interval goal, **LPC** < 1. Owing to the bad coin flip, from item 7, we obtain **LTER** < 1.

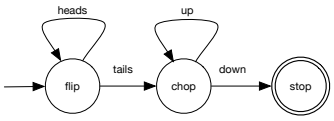

Figure 6: controller for flip and chop

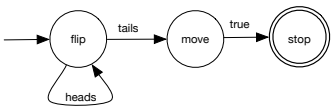

Figure 7: controller for flip and move

Theorem 30: *The 9 combinations of correctness and termination criteria identified are all distinct: that is, there are planning problems that belong to one and none other.*

7 {Reachability, Achievability} $\stackrel{?}{=}$ Adequacy

We build on the generality of our controller framework and relate to two influential accounts on plan correctness. The thrust of this section is to limit environments to the specific structural assumptions of those proposals, and show that these accounts are equivalent to some type of adequacy.

Goal Reachability

For the class of finite state systems with nondeterminism, Cimatti et al. (2003) defined three types of solutions: *weak*, *strong* and *strong cyclic*. In what follows, we first introduce their framework, and then discuss what these solutions mean in the context of adequacy.

Definition 31: (Cimatti et al. 2003) A *planning domain* $\mathcal{D}$ is a tuple $\langle \mathcal{X}, \mathcal{S}, \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{X}$ is a finite set of propositions, $\mathcal{S} \subseteq 2^{\mathcal{X}}$ is the set of states, $\mathcal{A}$ is a finite set of actions, and $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the transition relation. A *planning problem* $\mathcal{B}$ over $\mathcal{D}$ is a tuple $\langle \mathcal{D}, \mathcal{I}, \mathcal{G} \rangle$ where $\mathcal{I}, \mathcal{G} \subseteq \mathcal{S}$ are the initial and goal states respectively.

A *state-action table* π for $\mathcal{D}$ is a set of pairs $\langle s, \alpha \rangle$, where $s \in \mathcal{S}$ and $\alpha \in \mathcal{A}$ provided there is a s' such that $\mathcal{R}(s, \alpha, s')$.

The *execution structure* induced by π from $\mathcal{I}$ is a tuple $\mathcal{K} = \langle \mathcal{Q}, \mathcal{T} \rangle$, where $\mathcal{Q} \subseteq \mathcal{S}$ and $\mathcal{T} \subseteq \mathcal{S} \times \mathcal{S}$, which is the smallest set satisfying:

- if $s \in \mathcal{I}$ then $s \in \mathcal{Q}$
- if $s \in \mathcal{Q}$ and there is an action α and s' such that $\mathcal{R}(s, \alpha, s')$ then $s' \in \mathcal{Q}$ and $(s, s') \in \mathcal{T}$.

An *execution path* of $\mathcal{K}$ at $s_0 \in \mathcal{I}$ is a possibly infinite sequence $s_0 \cdot s_1 \cdot s_2 \cdots$ such that: either s_i is the last state of the sequence (that is, a *terminal state*) or $(s_i, s_{i+1}) \in \mathcal{T}$.

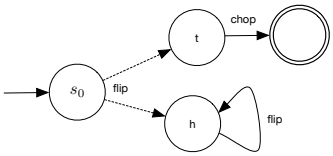

Figure 8: bad flip and chop problem

A planning problem $\mathcal{B}$ defined as above is essentially a generalized planning problem $\overline{\mathcal{P}}$: a fully observable and finite environment $\mathcal{E}$ is constructed from $\mathcal{S}, \mathcal{A}$ and $\mathcal{R}$ in an obvious way, and then we let $\overline{\mathcal{P}} = \{\langle s_0, \mathcal{E}, \mathcal{G}\rangle \mid s_0 \in \mathcal{I}\}$.

Cimatti et al. consider controller specifications by appealing to the state-action table π. In particular, suppose for every s mentioned in π, there is a unique α such that $\langle s, \alpha\rangle \in \pi$. Such a π is called *deterministic*, and it can be viewed as a controller in that it specifies the action to be performed at any given state.[4] Then, π is said to be a solution of a certain type (for $\mathcal{B}$) if the execution structure $\mathcal{K}$ satisfies appropriate properties, *e.g.* π is a weak solution if for any $s_0 \in \mathcal{I}$, some terminal state of $\mathcal{K}$ is reachable that is also in $\mathcal{G}$.

Putting it all together, we can establish the following:

Theorem 32: *Suppose $\mathcal{B}$ is as above, but with a deterministic state-action table, and suppose $\overline{\mathcal{P}}$ is the corresponding generalized planning problem. Then:*

- *weak solutions (for $\mathcal{B}$) are equivalent to* **ONE***-adequacy (in $\overline{\mathcal{P}}$);*
- *strong cyclic solutions are equivalent to* $\{\mathbf{PC}, \mathbf{TER}\}$*-adequacy;*
- *strong solutions are equivalent to* $\{\mathbf{ACYC}, \mathbf{PC}\}$*-adequacy.*

We can now leverage results from Section 3 and expand on these solution types. For example, in Theorem 16, we established that **BND** is implied by **ACYC**, and so:

Corollary 33: *Suppose $\mathcal{B}$ and $\overline{\mathcal{P}}$ are as in Theorem 32. Then, strong solutions imply* $\{\mathbf{BND}, \mathbf{PC}\}$*-adequacy.*

Theorem 16 also allows us to investigate the appropriateness of strong solutions in infinite domains:

Corollary 34: *Suppose $\mathcal{B}$ and $\overline{\mathcal{P}}$ are as in Theorem 32, except that the set of states $\mathcal{S}$ of $\mathcal{B}$ is infinite. Then, strong solutions do not imply* **BND***-adequacy.*

In sum, to capture the spirit of strong solutions in the sense of only admitting finite execution paths (or in our lingo: bounded runs), $\{\mathbf{PC}, \mathbf{BND}\}$-adequacy is better suited across finite and infinite planning domains.

[4]Note that this does not rule out nondeterministic transitions. It simply avoids ambiguities about which action is to be performed at a state. The case of state-action tables that are not deterministic is addressed in terms of all possible deterministic instances, which we omit here.

Goal Achievability

Building on (Levesque 1996), Lin and Levesque (1998), LL henceforth, considered the problem of *goal achievability*, which is perhaps best explained using an example. Imagine an agent facing two doors, the first of which has a tiger that can kill the agent and the second has gold. The agent, however, does not know which door has what. Although the goal of getting the gold is achievable by opening the second door, we would not say the agent *knows how* to get to the gold. To that end, LL define a controller framework to formalize achievability. Their work assumes binary sensing outcomes, and does not address nondeterministic outcomes.

Definition 35: (Lin and Levesque 1998) A history is an element of the set $\mathcal{R} = (\mathcal{A} \times \{0,1\})^*$, where $\{0,1\}$ is the set of observations, a controller $\mathcal{C}$ is any function mapping histories to $\mathcal{A} \cup \{stop\}$, and an environment $\mathcal{E}$ is any function mapping $\mathcal{R} \times \mathcal{A}$ to $\{0,1\}$. A controller is effective if $\mathcal{C}$ is recursive. A history σ is a run of the system $(\mathcal{C}, \mathcal{E})$ if inductively $\sigma = \epsilon$ is the empty sequence, or $\sigma = \sigma' \cdot (\alpha, \beta)$ where σ' is a run, $\mathcal{C}(\sigma') = \alpha \in \mathcal{A}$, and $\mathcal{E}(\sigma', \alpha) = \beta \in \{0,1\}$.

Basically, LL's notion of an environment can be seen a simplified account for noise-free environments, which implicitly changes after an action is performed by the controller. While histories in LL are different from ours in also mentioning the actions, this difference is not crucial: in any run of a system, actions mentioned in histories are precisely those advised by the controller.

Although controllers and environments are defined in an abstract fashion by LL, the notion of goal achievability is formalized in the logical language of the situation calculus (Reiter 2001). For this reason, we reformulate their intuitions in an abstract setting:

Definition 36: Suppose $\overline{\mathcal{P}} = \{\langle s_0, \mathcal{E}, \mathcal{G}\rangle \mid s_0 \in \mathcal{I}\}$ is a noise-free generalized planning problem over goal states $\mathcal{G}$. We say that $\mathcal{G}$ is (effectively) achievable iff there is a (effective) LL-controller $\mathcal{C}$ such that for every $\langle s_0, \mathcal{E}, \mathcal{G}\rangle \in \overline{\mathcal{P}}$, there is a terminating run σ at s_0 of $(\mathcal{C}, \mathcal{E})$ and $end(\sigma) \in \mathcal{G}$.

As a realization of effective controllers, LL consider *robot programs* (Levesque 1996) built from atomic actions $\mathcal{A}$, sequences, branches and loops, and they define what it means for these programs to achieve goals (omitted). A central result in their work is that a goal is effectively achievable iff there is a robot program that achieves the goal.

Putting it all together, we obtain the following result:

Theorem 37: *Suppose $\mathcal{O} = \{0,1\}$, and $\overline{\mathcal{P}}$ as above. A robot program achieves $\mathcal{G}$ iff there is an effective controller that is* **ONE***-adequate for $\overline{\mathcal{P}}$.*

Since the environment is noise-free, it is not surprising that **ONE**-adequacy is deemed sufficient; for example, as seen in Theorem 15, $\{\mathbf{PC}, \mathbf{TER}\}$-adequacy is implied, and by Theorem 14, so is $\{\mathbf{PC}, \mathbf{BND}\}$-adequacy.

8 Discussion and Related Work

In the previous sections, the notion of adequacy was shown to cover a range of planning scenarios. There is, however, one glaring omission from this discussion.

Perhaps the most influential planning framework in stochastic domains is the formulation of *Markov decision processes* (MDPs), a natural model for sequential decision making under Markovian assumptions on probabilistic transitions (Boutilier, Dean, and Hanks 1999). In brief, a solution to a MDP is a mapping from states to actions that maximizes the expected reward over an infinite history, where a discount factor favours current rewards over future ones. (See (Kučera and Stražovskỳ 2008), for example, for a synthesis framework, and (Chatterjee and Chmelík 2015) on infinite runs over rewards.) Although a number of recent approaches compute loopy plans for MDPs (Poupart and Boutilier 2004; Guestrin et al. 2003; Pajarinen and Peltonen 2011; Kumar and Zilberstein 2015), the use of these plans is quite different from our own. In particular, these plans usually do not involve a stopping action, and so the termination of runs is overlooked, and of course, there are no designated goal states (in the standard model (Geffner and Bonet 2013)). All of this precludes an immediate connection to adequacy criteria based on termination and goal satisfaction. An interesting possibility, then, is to appeal to formulations of reward functions as goal states (Koenig 1992; Geffner and Bonet 2013) and reason about correctness on the revised model. In the long term, it would be worthwhile to consider notions of adequacy that are determined by the expected reward for the course of action advised by the controller, and contrast terminating runs versus non-terminating runs with discount factors.

In a similar vein, as hinted in Section 7, the areas of knowledge-based planning and programming (Sardiña et al. 2006; Petrick and Bacchus 2004; Kaelbling and Lozano-Pérez 2013; Reiter 2001) is closely related to generalized planning (Levesque 1996). Intuitively, the initial state of a basic problem can be seen as a world considered epistemically possible by the agent, and so a generalized planning problem corresponds to a set of possible world states. A θ-adequate controller for a generalized planning problem can be seen to achieve θ in all the world states. In that regard, a generalized planning problem as considered here treats knowledge categorically. In robotic applications, the agent entertains a probability distribution on the possible worlds, and it may be unrealistic (and unnecessary) to compute controllers that are adequate for all worlds. It is then desirable to develop an epistemic account of adequacy based on *degrees of belief* (Belle and Levesque 2013; 2015). This would allow, for example, the robot to believe that a controller is adequate with a high degree of certainty. We believe that such an account would further relate goal-based adequacy and (partially observable) MDPs.

We now cover prior work in the area. We will briefly touch on approaches to generating loopy plans, and then discuss related correctness concerns.[5] At the outset, we remark that the relationships between various kinds of termination and goal-based correctness criteria in unbounded domains, and their probabilistic and continuous versions, have not been discussed in a formal and general way elsewhere.

[5]For a review of sequential and conditional planning, see (Geffner and Bonet 2013).

Early approaches to loopy plans can be seen as deductive methodologies, often influenced by program synthesis and correctness (Hoare 1969). Manna and Waldinger (1980) obtained recursive plans by matching induction rules, and Stephan and Biundo (1996) refine generic plan specifications, but required input from humans. See (Magnusson and Doherty 2008) for a recent approach using induction.

Most recent proposals differ considerably from this early work using deduction:

- Levesque (2005) expects two parameters with the planning problem; the approach plans for the first parameter, winds it to form loops and tests it for the second.
- Winner and Veloso (2007) synthesize a plan sequence with partial orderings, and exploit repeated occurrences of subplans to obtain loops.
- Srivastava (2010) considers an abstract state representation that groups objects into equivalences classes, the idea being that any concrete plan can be abstracted wrt these classes and repeated occurrences of subplans can be leveraged to generate compact loopy plans.
- Bonet, Palacios, and Geffner (2009) integrate the dynamics of a memoryless plan with a planning problem, and convert that to a conformant planning problem; the solution to this latter problem is shown to generalize to multiple instances of the original problem.
- Hu and De Giacomo (2013) propose a bounded AND/OR search procedure that is able to synthesize loopy plans.

On the matter of correctness, Levesque (1996) argued that generalized plans be tested for termination and correctness against all problem instances; Lin and Levesque (1998) extended this account to define goal achievability. In later work, Cimatti et al. (2003) defined the notions of weak, strong and strong cyclic solutions in the presence of nondeterminism.[6] These notions are widely used in the planning community (Bonet and Geffner 2015); see, for example, Bertoli et al. (2006) for an account of strong planning with a sensor model. Recently, Srivastava et al. (2015) synthesize loopy plans in domains with nondeterministic quantitative effects, for which strong cyclic solutions are studied.

Beyond a definition for the correctness of plans, what is also needed are techniques that can automatically verify that a plan is correct for a specification. In that regard:

- Hu and Levesque (2011) show how termination and correctness can be checked for certain classes of problems purely based on the syntax of the problem specification.
- Lin (2008) considers conditions under which a finite set of models of the planning problem, formulated as a logical theory, is sufficient for proving goal achievability.
- Srivastava (2010) shows that certain domains can be reduced to register machines, which are then amenable to effective verification.

[6]Variant additional stipulations in the literature include things like *fairness*, where every outcome of a nondeterministic action must occur infinitely often (Bonet and Geffner 2015).

9 Conclusions

This paper investigated the adequacy of generalized plans, such as plans with loops, for domains that are possibly unbounded, and/or stochastic, and/or continuous. A general controller framework was used to show the relationships between adequacy notions motivated from first principles. The idea is that a plan can be argued to be correct for a planning problem in terms of appropriate notions. Finally, we saw examples of plans that meet specifications, and that categorical adequacy may not be possible in certain problems.

Many of the results investigated in this paper use simple plans, not unlike the ones synthesized in the literature. This raises a number of questions and avenues for future work:

- How can we synthesize loopy plans for unbounded stochastic domain using an existing system for noise-free domains? Can qualitative specifications of stochastic ones be devised that would allow us to leverage existing work?
- Along the lines of (Srivastava 2010; Hu and Levesque 2011), how can we automatically verify that a given loopy plan is correct for unbounded stochastic domains?

In the long term, relating expected rewards, noisy sensing, belief-based planning and adequacy is a promising direction for investigating foundational questions, especially in service of unifying loopy plan frameworks from the goal-based and decision-theoretic camps.

References

Belle, V., and Levesque, H. J. 2013. Reasoning about continuous uncertainty in the situation calculus. In *IJCAI*.

Belle, V., and Levesque, H. J. 2015. Allegro: Belief-based programming in stochastic dynamical domains. In *IJCAI*.

Bertoli, P.; Cimatti, A.; Roveri, M.; and Traverso, P. 2006. Strong planning under partial observability. *Artificial Intelligence* 170(4–5):337 – 384.

Biundo, S. 1994. Present-day deductive planning. In *Current Trends in AI Planning. EWSP*, 1–5.

Bonet, B., and Geffner, H. 2015. Policies that generalize: Solving many planning problems with the same policy. In *IJCAI*.

Bonet, B.; Palacios, H.; and Geffner, H. 2009. Automatic derivation of memoryless policies and finite-state controllers using classical planners. In *ICAPS*.

Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research* 11(1):94.

Camacho, A.; Muise, C.; Ganeshen, A.; and McIlraith, S. A. 2015. From fond to probabilistic planning: Guiding search for quality policies. In *Workshop on Heuristic Search and Domain Independent Planning, ICAPS*.

Chatterjee, K., and Chmelík, M. 2015. Pomdps under probabilistic semantics. *Artificial Intelligence* 221:46–72.

Cimatti, A.; Pistore, M.; Roveri, M.; and Traverso, P. 2003. Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence* 147(1–2):35 – 84.

Geffner, H., and Bonet, B. 2013. *A Concise Introduction to Models and Methods for Automated Planning*. Morgan and Claypool Publishers.

Guestrin, C.; Koller, D.; Gearhart, C.; and Kanodia, N. 2003. Generalizing plans to new environments in relational MDPs. In *IJCAI*.

Hoare, C. A. R. 1969. An axiomatic basis for computer programming. *Commun. ACM* 12(10):576–580.

Hopcroft, J. E., and Ullman, J. D. 1979. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley Publishing Company.

Hu, Y., and De Giacomo, G. 2013. A generic technique for synthesizing bounded finite-state controllers. In *ICAPS*.

Hu, Y., and Levesque, H. J. 2011. A correctness result for reasoning about one-dimensional planning problems. In *IJCAI*, 2638–2643.

Kaelbling, L. P., and Lozano-Pérez, T. 2013. Integrated task and motion planning in belief space. *I. J. Robotic Res.* 32(9-10):1194–1227.

Koenig, S. 1992. Optimal probabilistic and decision-theoretic planning using markovian decision theory. Technical Report UCB/CSD-92-685, EECS Department, University of California, Berkeley.

Kučera, A., and Stražovskỳ, O. 2008. On the controller synthesis for finite-state markov decision processes. *Fundamenta Informaticae* 82(1-2):141–153.

Kumar, A., and Zilberstein, S. 2015. History-based controller design and optimization for partially observable MDPs. In *ICAPS*.

Levesque, H. J. 1996. What is planning in the presence of sensing? In *AAAI / IAAI*, 1139–1146.

Levesque, H. 2005. Planning with loops. In *IJCAI*, 509–515.

Lin, F., and Levesque, H. J. 1998. What robots can do: Robot programs and effective achievability. *Artificial Intelligence* 101(1-2):201–226.

Lin, F. 2008. Proving goal achievability. In *KR*, 621–628.

Magnusson, M., and Doherty, P. 2008. Deductive planning with inductive loops. In *KR*, 528–534.

Manna, Z., and Waldinger, R. J. 1980. A deductive approach to program synthesis. *ACM Trans. Program. Lang. Syst.* 2(1):90–121.

Matarić, M. J. 2007. *The robotics primer*. Mit Press.

Pajarinen, J. K., and Peltonen, J. 2011. Periodic finite state controllers for efficient pomdp and dec-pomdp planning. In *NIPS*, 2636–2644.

Petrick, R., and Bacchus, F. 2004. Extending the knowledge-based approach to planning with incomplete information and sensing. In *ICAPS*, 2–11.

Poupart, P., and Boutilier, C. 2004. Bounded finite state controllers. In *NIPS*, 823–830.

Reiter, R. 2001. *Knowledge in action: logical foundations for specifying and implementing dynamical systems*. MIT Press.

Sardiña, S.; De Giacomo, G.; Lespérance, Y.; and Levesque, H. J. 2006. On the limits of planning over belief states under strict uncertainty. In *KR*, 463–471.

Srivastava, S.; Zilberstein, S.; Gupta, A.; Abbeel, P.; and Russell, S. 2015. Tractability of planning with loops. In *AAAI*.

Srivastava, S. 2010. *Foundations and Applications of Generalized Planning*. Ph.D. Dissertation, Department of Computer Science, University of Massachusetts Amherst.

Stephan, W., and Biundo, S. 1996. Deduction-based refinement planning. In *AIPS*, 213–220.

Thrun, S.; Burgard, W.; and Fox, D. 2005. *Probabilistic Robotics*. MIT Press.

Winner, E., and Veloso, M. M. 2007. LoopDISTILL: Learning domain-specific planners from example plans. In *Workshop on AI Planning and Learning, ICAPS*.